

General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.
- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.
- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.
- This document is paginated as submitted by the original source.
- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

TECHNICAL MEMORANDUM (NASA) 61

LORAN-C FLIGHT TEST SOFTWARE

Described is the software package developed for the KIM-1 Micro-System and the Mini-L PLL receiver to simplify taking flight test data at Ohio University.

by

James D. Nickum

Avionics Engineering Center
Department of Electrical Engineering
Ohio University
Athens, Ohio 45701

August 1978



Supported by

National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia
Grant NGR 36-009-017

TABLE OF CONTENTS

	<u>PAGE</u>
I. INTRODUCTION	1
II. ADDRESS BUFFER HARDWARE	1
III. WORD GENERATOR HARDWARE AND TIMING	1
IV. FLIGHT TEST SOFTWARE	5
V. SUMMARY	5
VI. REFERENCES	14
VII. APPENDICES	15
Appendix A	16
Appendix B	17
Appendix C	33

I. INTRODUCTION

This technical memorandum will describe the flight test software for use with the Mini-L LORAN-C Receiver. In order to describe completely the software programs the interface hardware and timing are also presented. Also included is the description of the address and data bus buffers used in the KIM-1 Micro-system^[1].

II. ADDRESS BUFFER HARDWARE

The address and data buffers (ADB) are buffers that allow the easy expansion of the KIM-1 Microcomputer System. Referring to Figure 1, the data bus is buffered using two 8833 bidirectional data buffers. The 7400 logic gates provide the necessary read/write steering logic necessary since the data bus is bidirectional. The 4 diodes provide the decoding for a high impedance level on the KIM-1 data bus during all times when the KIM-1 is accessing any internal (KIM-1) memory or peripheral. This is necessary since the ADB described here are between the outside world and the KIM-1, and not the CPU chip and the outside world.

The address bus buffers employ 3 (three) 8097 tri-state buffers. Since the address bus is an output-only bus, these devices are enabled at all times.

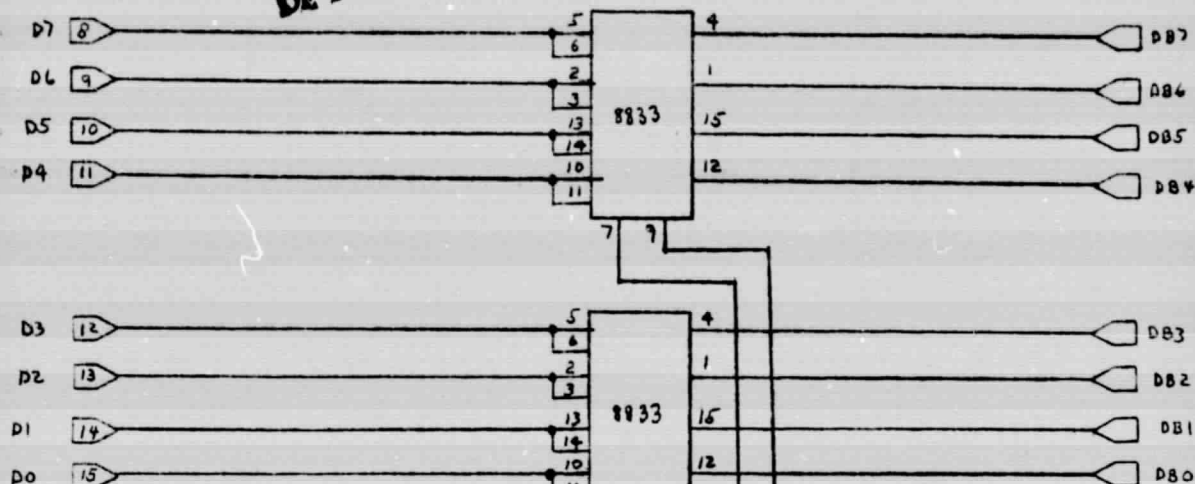
III. WORD GENERATOR HARDWARE AND TIMING

The basic word generator circuit is described in detail in NASA TM-54. The circuit described here differs somewhat with regard to the signal inputs and signal outputs from the one described in NASA TM-54^[2]. The basic purpose of the word generator is to clock a counter at a 1 MHz rate and reset it at the GRI rate. If the output pulse from the Mini-L LORAN-C Receiver is from the master loop, then the word generator counter is cleared to zero. If it is from one of the slave loops the value of the counter is clocked into the word latch, and, therefore, the KIM-1 can read the measured time in microseconds. Refer to Figure 2, the timing logic, and Figure 3, the timing diagram, to describe the operation. The Master, Slave 1 and Slave 2 are treated equally with regard to the interrupt generation. Pulses are ORed to the D input of U16, which generates a 500 ns. pulse LTIME, which is used to clock the latch. Device U17 forms the interrupt request to the KIM-1. The D flip flop U27 catches and one-shots the master output and the latch made up of U25 provides a test bit MTB to the software to determine the master. The CNTLD signal is generated by U27 and this is used to clear the counters on the occurrence of the master. Input LOREN is a LORAN enable output flag for software control of hardware interrupts from the word generator to the KIM-1. The SIRQ and LIRQ are interrupt inputs that were used for earlier versions of the Mini-L receivers. The DECODE 0 is generated by software and occurs whenever a read of the MSB of the GRI word generator is made. This is used as a hardware reset as the normal reading of the word generator is from LSB to MSB; therefore, when the MSB is

ORIGINAL
DE POOR QUALITY

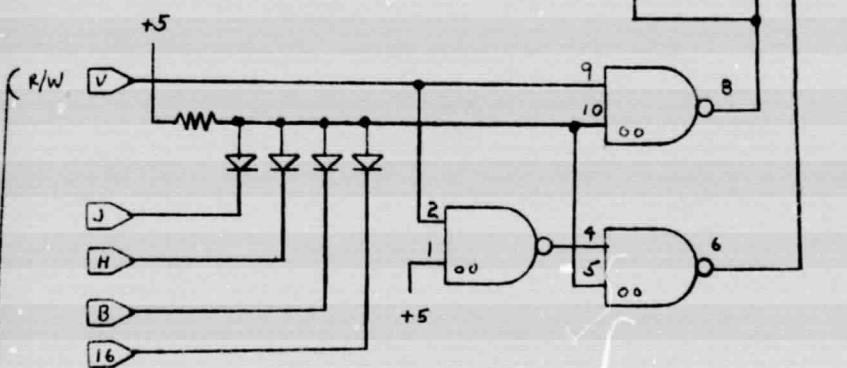
FROM

KIM-1 APPLICATION

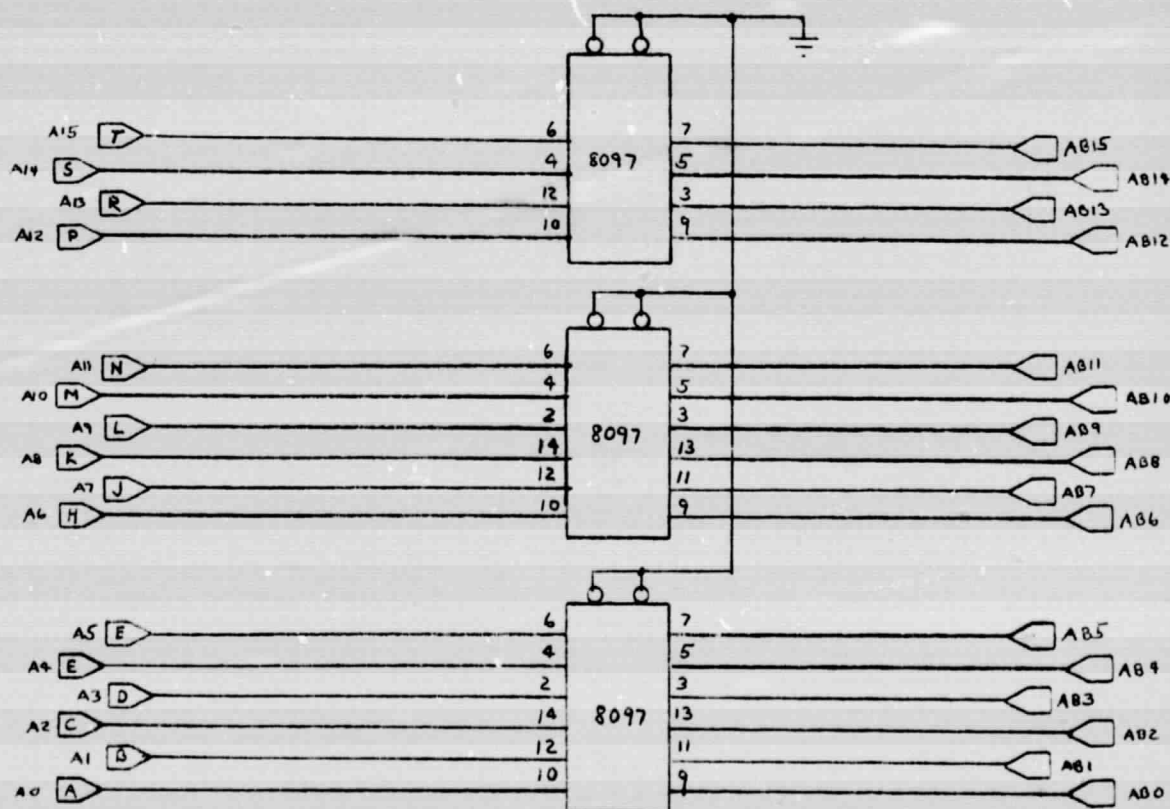


DATA

MICRO SYSTEM BUSES



FROM
KIM-1 EXPANSION



ADDRESS

Figure 1. Micro-System Bus Buffers.

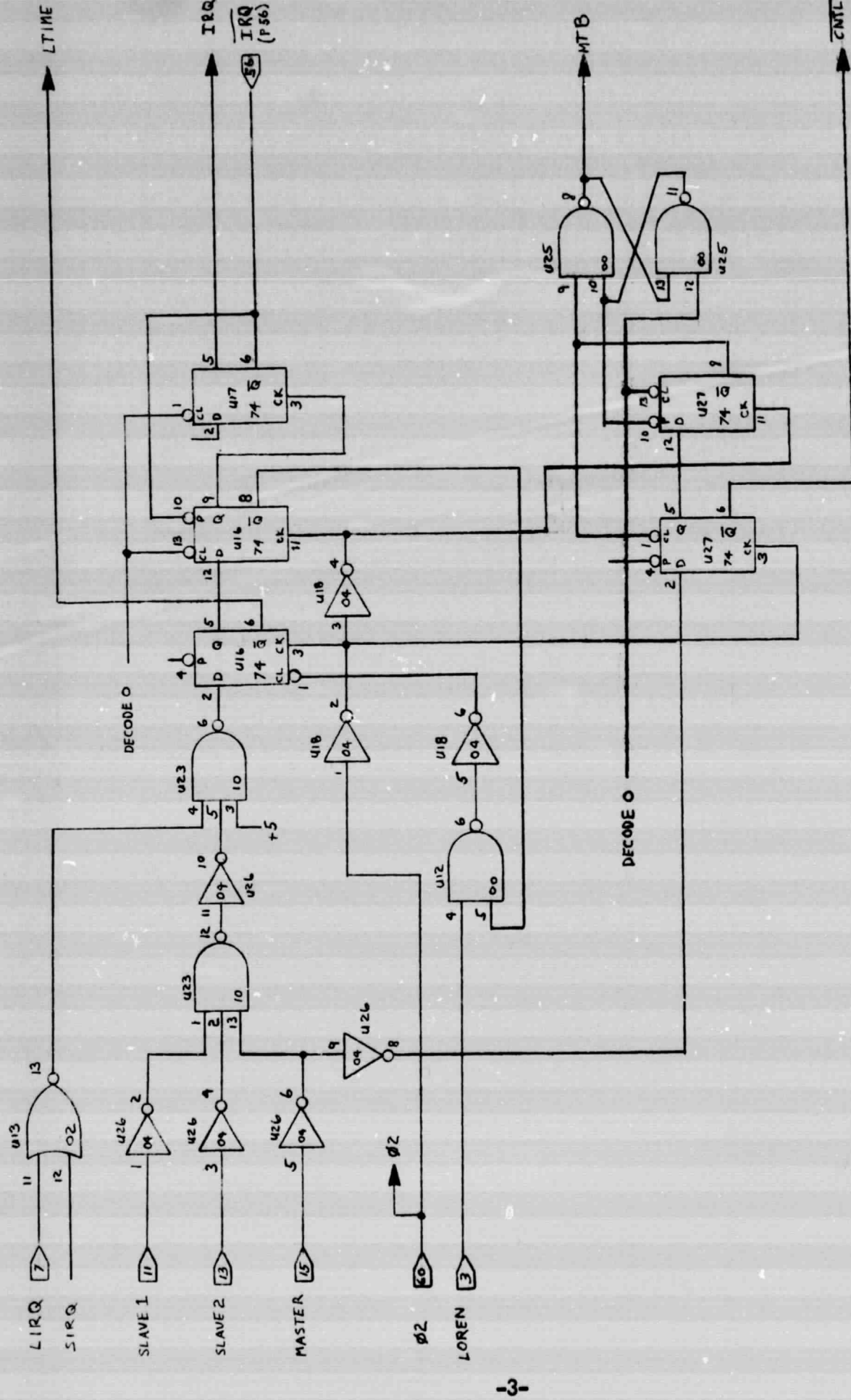
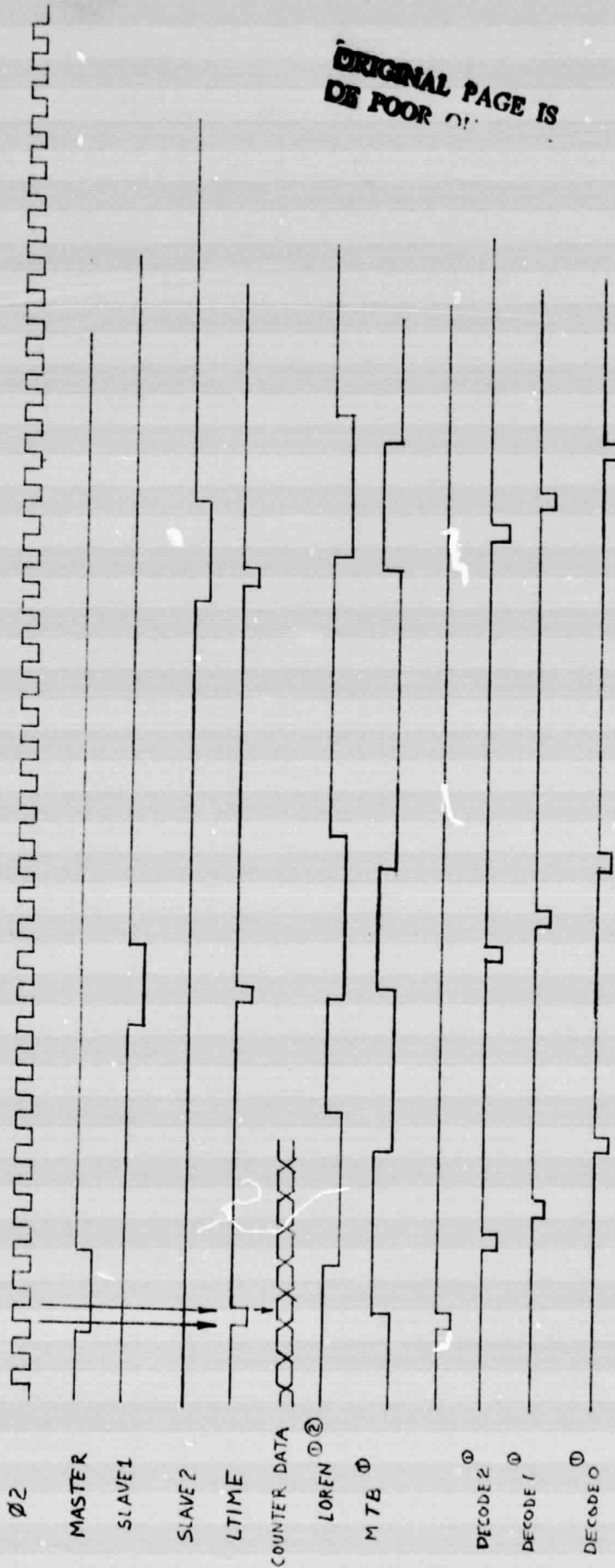


Figure 2. Word Generator Timing Logic.

ORIGINAL PAGE IS
OF POOR QUALITY



- ① NOT TO SCALE
- ② SOFTWARE GENERATED

Figure 3. Word Generator Timing Diagram.

read the word generator is reset. Figure 4 is the schematic of the counter and latches for the word generator. The counters are clocked on the leading edge of the 1 MHz clock $\Phi 2$. The latches are clocked coincident with the trailing edge of $\Phi 2$ by LTIME. This allows the counter outputs to stabilize before the contents of the counters are latched. Also provided are an input and output port, U28 for input and U19 and U20 for output. These ports provide sense and flag capabilities to the software program. Figure 5 is the address decoder for the word generator. It provides all of the necessary address decoding for the various functions. It also provides a memory select to the KIM-1 to provide a memory disable for bus access.

IV. FLIGHT TEST SOFTWARE

Figure 6 shows the initialization flow chart. This part of the program sets up the word generator hardware. It also loads the interrupt vector locations, clears and sets all software data and flag locations. The background software is entered directly. Figure 7 is the LIRQ interrupt service routine. This routine determines whether the interrupt came from a master or a slave loop and also keeps track of the average values for the time differences. Essentially this involves adding the time differences of ten measurements, rounding the next to the last digit and shifting right one digit to form an average of ten measurements. Also this routine counts every 10th GRI to form a real time clock based on LORAN-C.

Figure 8 is the flow diagrams for the background processing. This processing occurs continuously between interrupts. Its function is to receive and process appropriate commands from the keyboard that determine what information is to be displayed or stored. The functions that are provided are the display of either of the time differences or the number of data points stored. In addition, the software will automatically multiplex out to a Rustrak chart recorder the last two significant digits of both time differences. A pilot's guidance display (PGD) is provided and either LOP can be selected to be tracked by the pilot. This PGD provides a 20 μ s left right steering information to the pilot. Figure 10 is the flow diagram for the NMI interrupt service routine. This routine simply sets a flag that is sensed in the background processing routine. This causes the current value of the GRI clock counter and the two time differences to be stored in RAM. This allows for the collection of data during the flight.

V. SUMMARY

The software and hardware described here are provided to allow an easy and efficient way to collect and display data relating to LORAN-C position fixes generated by the Mini-L receiver. The software provided here has been designed and structured so that modifications and additions can be implemented relatively easily.

There are several subroutines used and they are documented along with a complete program listing of the flight test software in the Appendix. Also provided are instructions for the use of the program.

ORIGINAL PAGE IS
OF POOR QUALITY.

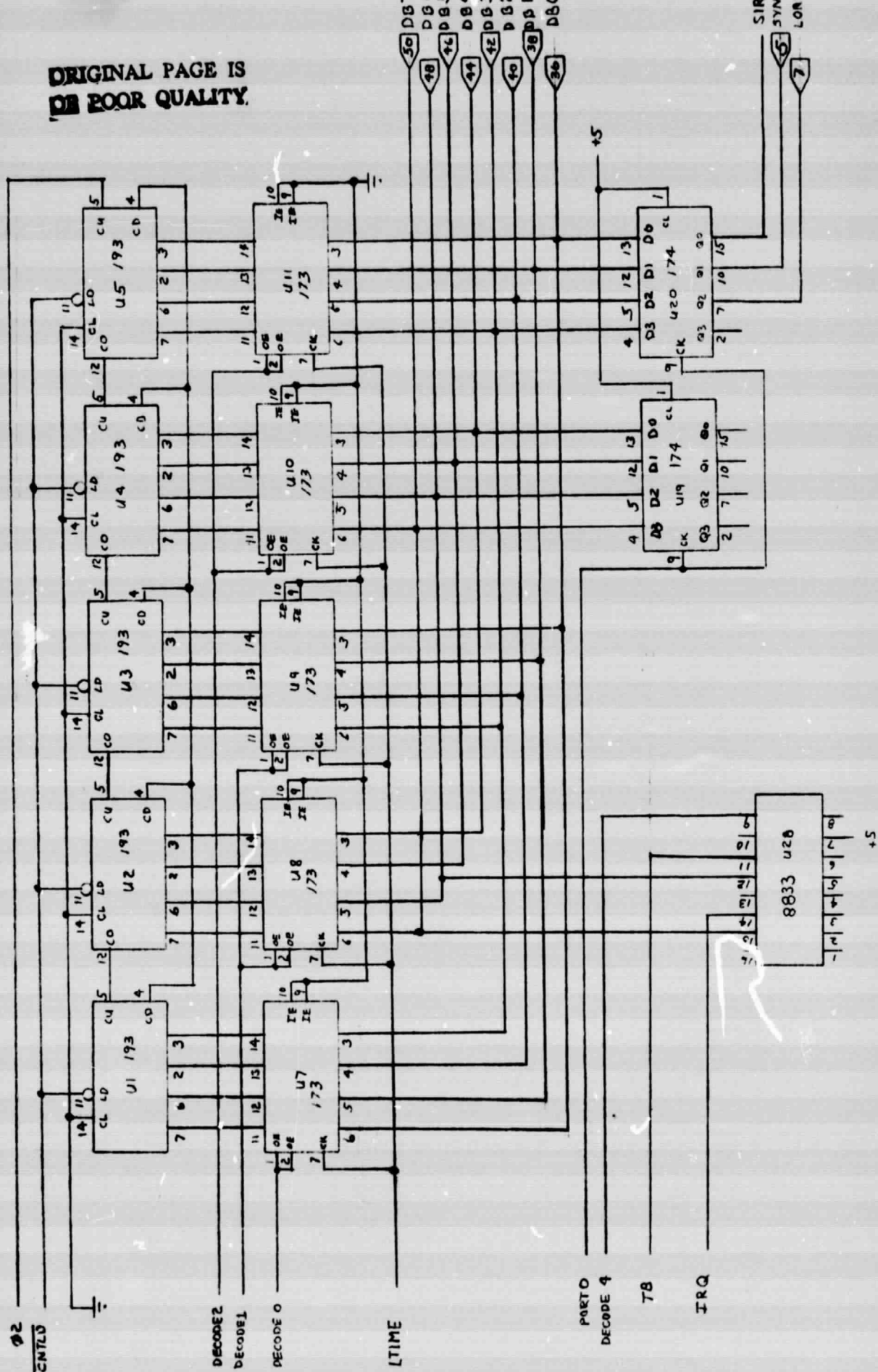


Figure 4. Word Generator Counter-Latch Schematic.

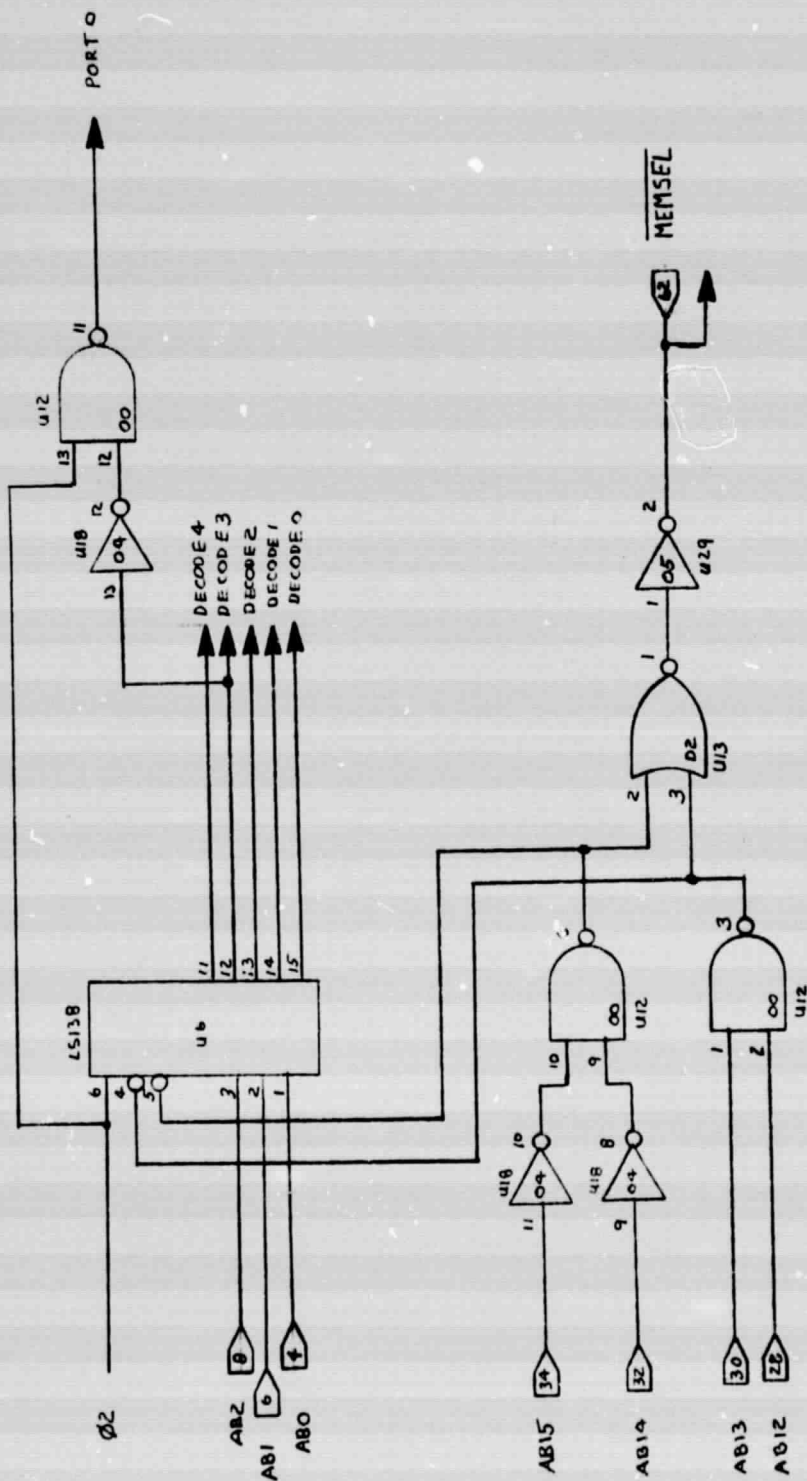


Figure 5. Word Generator Address Decoding.

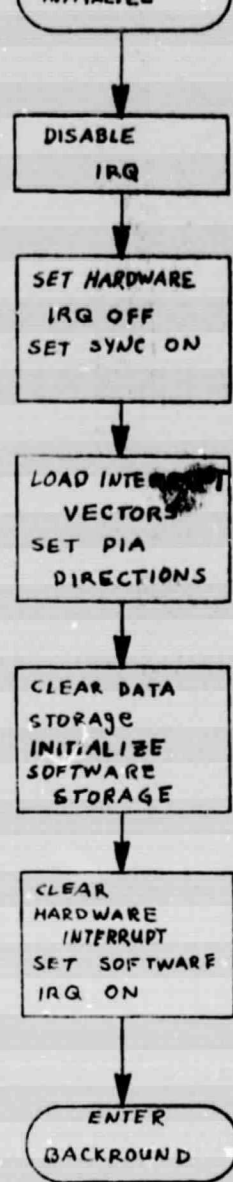


Figure 6. Initialization Flow Diagram.

ORIGINAL PAGE IS
OF POOR QUALITY

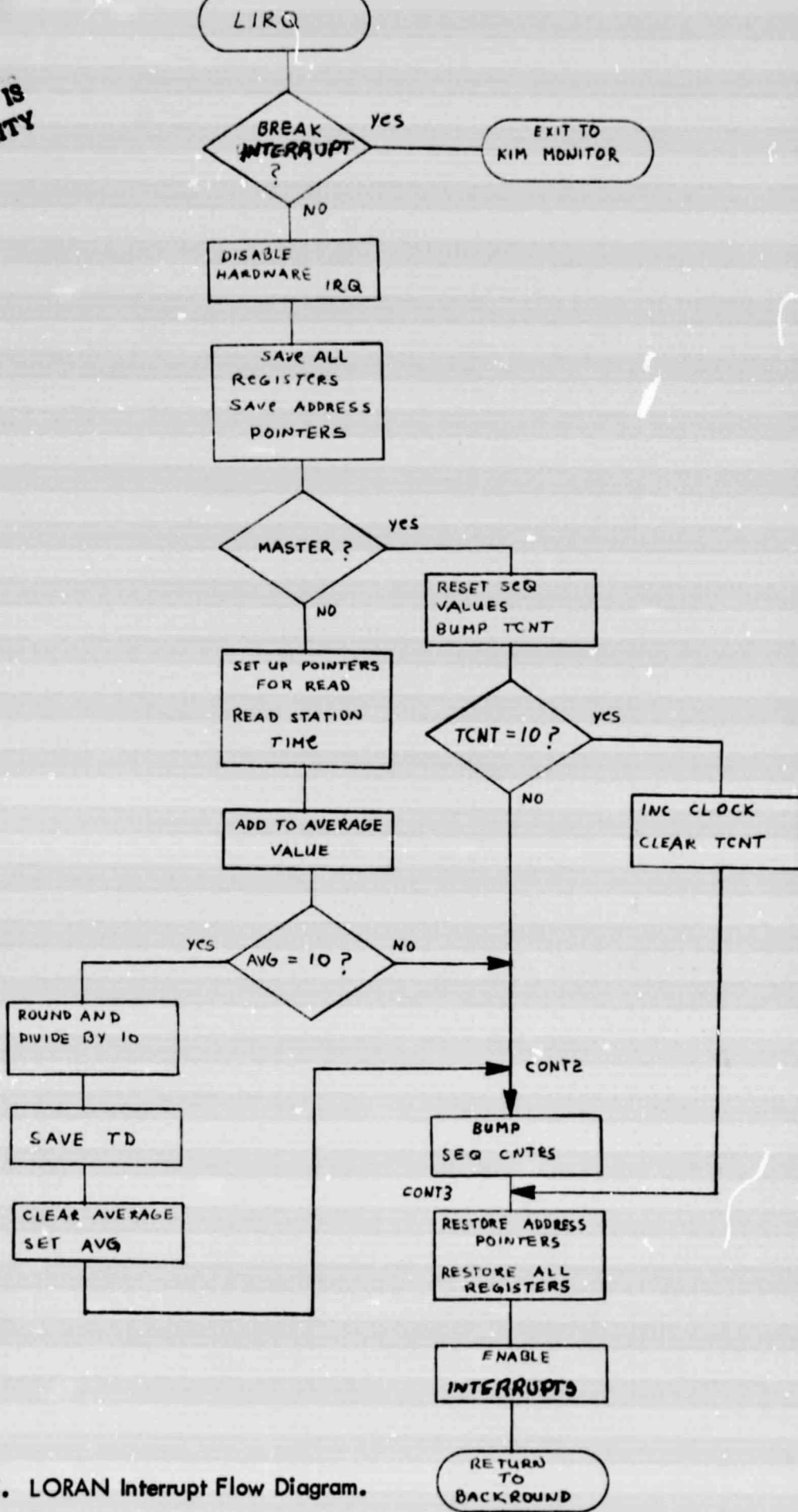


Figure 7. LORAN Interrupt Flow Diagram.

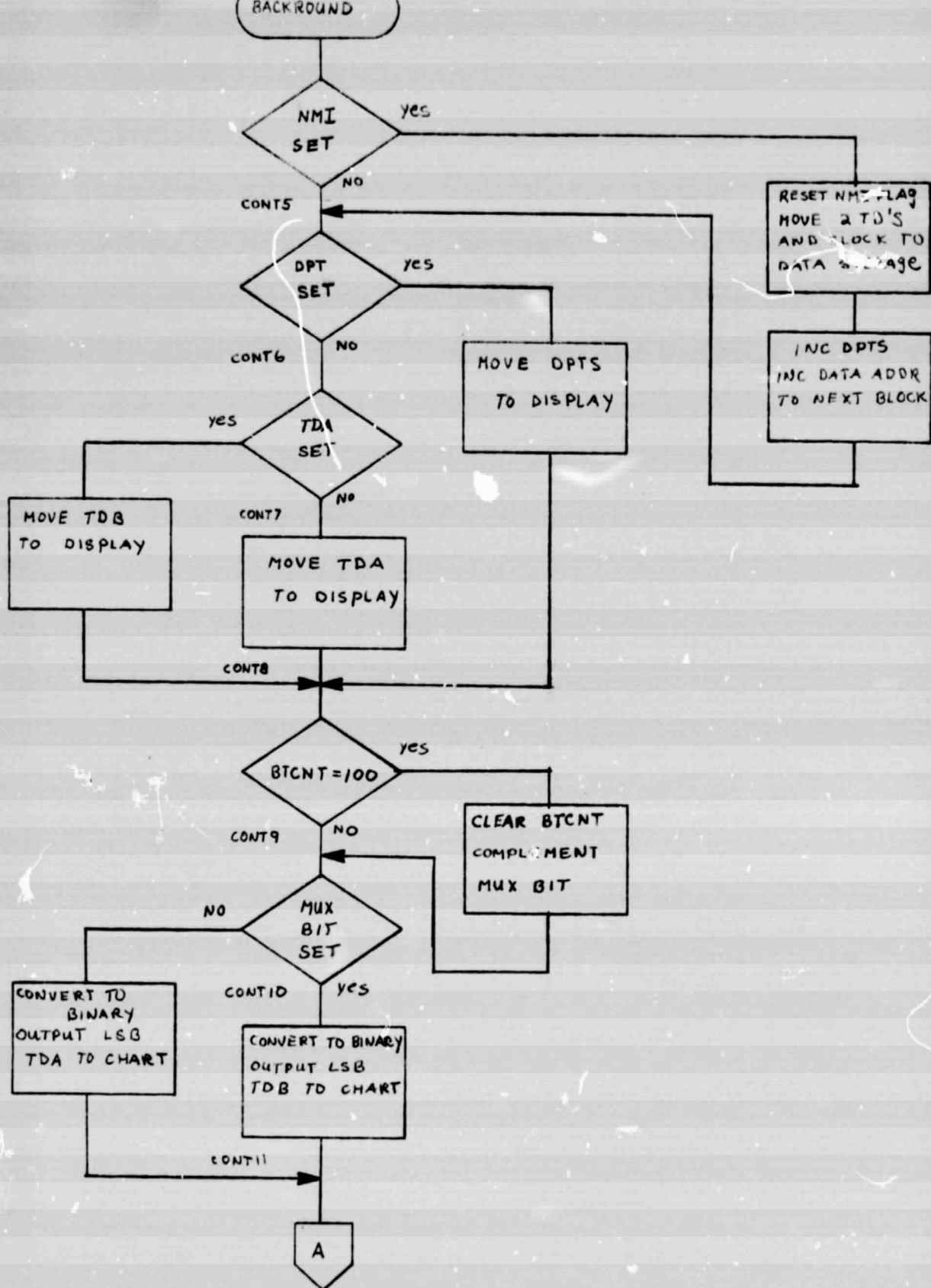


Figure 8a. Background Processing Flow Diagram.

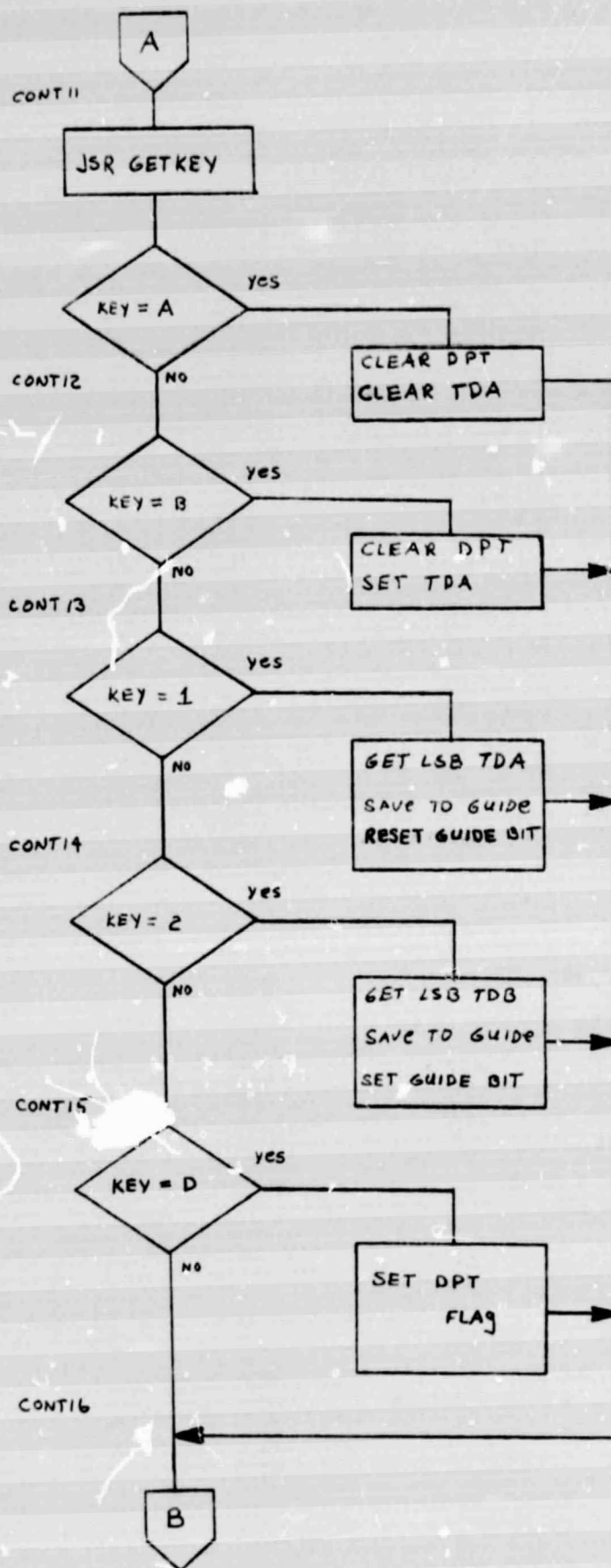


Figure 8b. Background Flow Diagram (Continued).

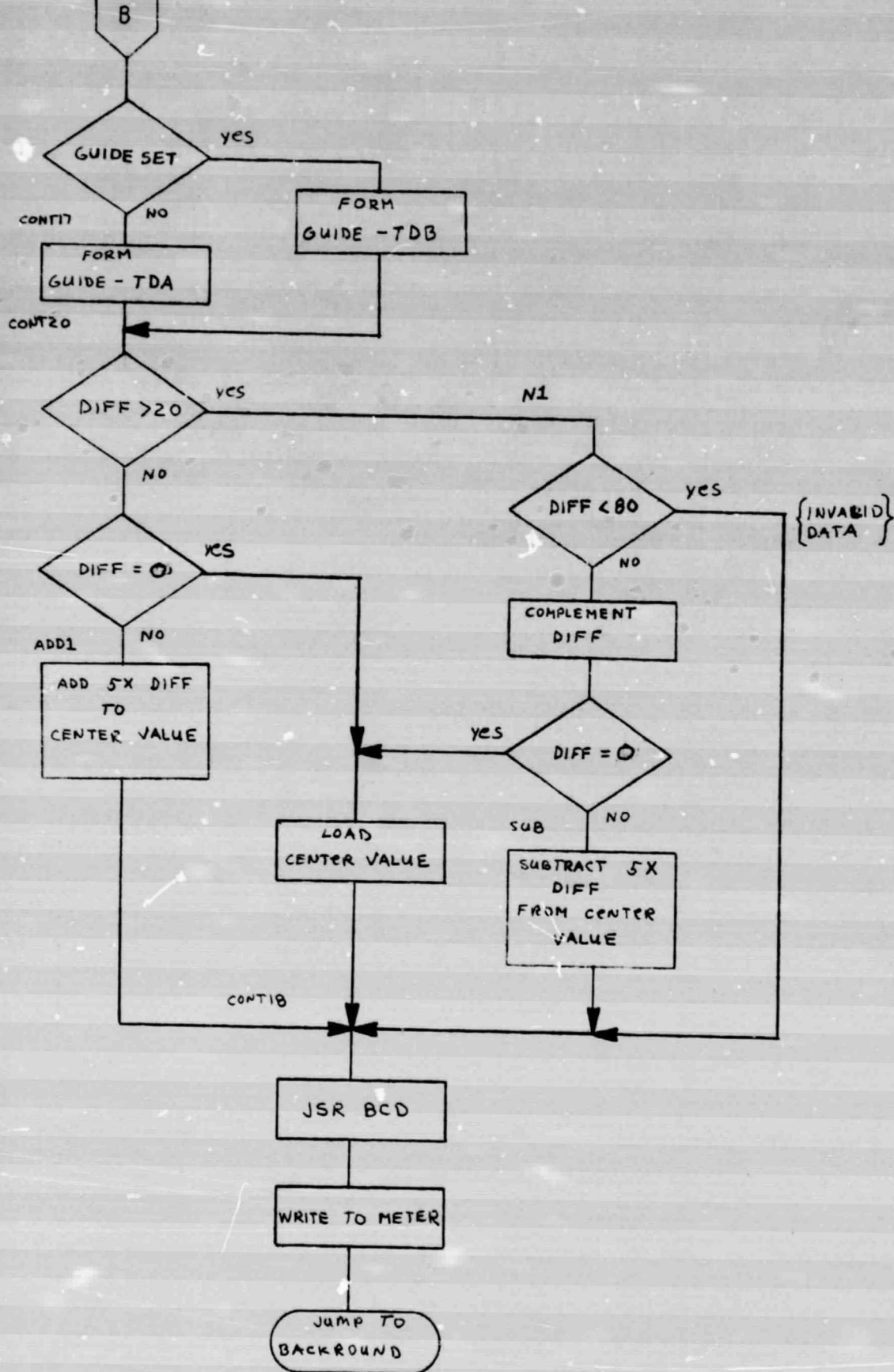


Figure 8c. Background Flow Diagram (Continued).

ORIGINAL PAGE IS
OF POOR QUALITY



Figure 9. NMI Interrupt Flow Diagram.

In summary, the flight test software and hardware provided here allows the user to display both LOP time differences. It provides storage for data points that contain both LOP's and an elapsed GRI count. It also provides a display of the number of data points stored, and finally it provides a pilot's steering command from either LOP.

VI. REFERENCES

- [1] Nickum, James, "Stand-Alone Development System Using a KIM-1 Microcomputer Module", NASA Technical Memorandum No. 56, Avionics Engineering Center, Department of Electrical Engineering, Ohio University, Athens, Ohio, March 1978.
- [2] Nickum, James, "LORAN-C Digital Word Generator for Use with a KIM-1 Microprocessor System", NASA Technical Memorandum No. 54, Avionics Engineering Center, Department of Electrical Engineering, Ohio University, Athens, Ohio, December 1977.

VII. APPENDICES

APPENDIX A

A. Flight Test Software Operation. From cassette tape the program is loaded in two parts. The first part loads locations 0000-0050 Hex. The second part loads locations 0200-050F Hex. The starting address is 0204 Hex. The interrupt command cable from the Mini-L to the KIM-1 Micro-System should be in place before starting.

To begin the program load 0204 Hex into the address on the KIM-1 display. Verify that the IRQ switch on the KIM-1 Micro-System is in the off position. Press GO; the display should show all zeros. If this is not the case, reload the program and start again. To synchronize the software GRI clock with a known time, turn on the IRQ switch on the KIM-1 Micro-System at a known time. About 1 second after the IRQ switch is turned on the display will show the LOP #1 time difference.

B. Data Request Operation. To display LOP #1 time difference press the "A" key on the Keyboard. To display the LOP #2 time difference press the "B" key. To display the number of used data points press the "D" key. Pressing the "ST" key will save the GRI clock, LOP #1 and LOP #2 to the data RAM for later readout. Each press of the "ST" key will cause this data to be stored in consecutive memory locations in RAM.

To cause the pilot's guidance display to provide guidance with respect to LOP #1 press the "1" key. To cause the same for LOP #2 press "2".

The software is self-synchronizing and provided that the Mini-L power or the KIM-1 Micro-System power does not fail the software will run continuously. To stop the software requires a computer reset. If a reset is necessary, then before starting the program again any data that has been recorded in the RAM data memory starting at 0E00 Hex must be saved, as the initialize routine in the flight test software will clear all data to zeros. Record the start time for later data analysis.

C. Key Codes for Flight Test Software.

<u>Key</u>	<u>Description of Function</u>
A	Displays value of LOP #1 in real time.
B	Displays value of LOP #2 in real time.
1	Causes the pilot display to provide steering commands with respect to the value of LOP #1 at the time it is presented.
2	Same as 1 except LOP #2 provides steering.
D	Displays the number of data points already stored.
ST	This key causes the current value of the GRI clock, LOP #1 and LOP #2 to be stored to RAM.

APPENDIX B . Software Listing.

END PASS 1: 0 ERRORS

ORIGINAL PAGE IS
OF POOR QUALITY

THIS PROGRAM WILL PROVIDE THE NECESSARY SOFTWARE TO SUPPORT A FLIGHT TEST OF THE MINI- L PLL RECIEVER. IT PERFORMES MANY FUNCTIONS FOR BOTH MONITORING THE PROGRESS OF THE FLIGHT AND ALSO RECORDING SPECIFIC DATA RELATED TO THE PATH OF THE FLIGHT WITH REGARD TO THE OBSERVED LORAN C TIME DIFFERENCES. IT CAN ALSO PROVIDE A RELATIVE MEASURE OF THE TIME OF OCCOURANCE OF A PARTICULAR EVENT RELATIVE TO THE GRI OF THE LORAN STATIONS BEING TRACKED. THE FOLLOWING IS A DEFINITION OF THE INPUT OUTPUT PORTS FOR THE PROGRAM. ALSO INCLUDED ARE SOFTWARE FLAG DEFINITIONS.

HCNTRL
\$3003

I	I	I	I	I	I	I	I	I	I
7	6	5	4	3	2	1	0		

2 LOREN
1 SYNCEN

MTB
\$3005

I	I	I	I	I	I	I	I	I	I
7	6	5	4	3	2	1	0		

7 MASTER TEST BIT

DFLAG
SOFTWARE

I	I	I	I	I	I	I	I	I	I
7	6	5	4	3	2	1	0		

7 NMI
6 TDA
5 MUX
4 DPT
3 GUIDE

INPUT
\$3000 LSBYTE
\$3001 NSBYTE
\$3002 MSBYTE

I	I	I	I	I	I	I	I	I	I
7	6	5	4	3	2	1	0		

```

53      *
54      *
55      *
56      *
57      *      THIS IS THE SYMBOLS DEFINITIONS FOR THE
58      *      PROGRAM
59      *
60      *
61      0000      SCANDS EQU $1F1F
62      0000      GETKEY EQU $1F6A
63      0000      INPUT EQU $3000
64      0000      DPIA1 EQU $1701
65      0000      DPIA2 EQU $1703
66      0000      METER EQU $1700
67      0000      CHART EQU $1702
68      0000      IRQL EQU $17FE
69      0000      NMIL EQU $17FA
70      0000      IOSO EQU $00000110
71      0000      IOSX EQU $00000100
72      0000      IXSO EQU $00000010
73      0000      IXSX EQU $00000000
74      0000      DISP EQU $00F9
75      0000      SAVEA EQU $00F3
76      0000      SAVEP EQU $00F1
77      0000      SAVEX EQU $00F5
78      0000      SAVEY EQU $00F4
79      0000      SAVE1 EQU $1C05
80      0000      XINPUT EQU 02
81      0000      XCLK EQU 04
82      0000      XTDA EQU 06
83      0000      XTDB EQU 08
84      0000      XDISP EQU $0A
85      0000      XBINPT EQU $0C
86      0000      XBTD3 EQU $0E
87      0000      XVALU1 EQU $10
88      0000      XVALU2 EQU $12
89      0000      XBVAL1 EQU $14
90      0000      XBVAL2 EQU $16
91      0000      XDATA EQU $18
92      0000      XAVGA EQU $1A
93      0000      XAVGB EQU $1C
94      0000      MTR EQU $3004
95      0000      HCNTRL EQU $3003
96      *
97      *
98      *      THIS IS THE BEGINNING OF THE ZERO PAGE
99      *      CONSTANTS, TEMP STORAGE AND ADDRESSING
100     *      POINTERS.
101     *
102     *
103     0000      CRG 0000
104     0000 00 00      PORGIN BSS 2
105     0002 00      AINPUT HEX 00,30
106     0003 30
107     0004 20      ACLK HEX 20,00

```

```

0005 00
107 0006 23      ATDA  HEX 23,00
0007 00
108 0008 26      ATDB  HEX 26,00
0009 00
109 000A F9      ADISP  HEX F9,00
000B 00
110 000C 29      ABINPT HEX 29,00
000D 00
111 000E 2C      ABTDB  HEX 2C,00
000F 00
112 0010 00 00    VALUE1 BSS 2
113 0012 00 00    VALUE2 BSS 2
114 0014 00 00    BVALU1 BSS 2
115 0016 00 00    BVALU2 BSS 2
116 0018 00      ADATA  HEX 00,12
0019 12
117 001A 2F      AAVGA  HEX 2F,00
001B 00
118 001C 32      AAVGB  HEX 32,00
001D 00
119 0020          ORG $0020
120 0020 00 00 00 CLK  BSS 3
121 0023 00 00 00 TDA  BSS 3
122 0026 00 00 00 TDB  BSS 3
123 0029 00 00 00 BINPT BSS 3
124 002C 00 00 00 BTDB  BSS 3
125 002F 00 00 00 AVGA  BSS 3
126 0032 00 00 00 AVGB  BSS 3
127 0035 00 00 00 VALUE3 BSS 3
128 0038 00 00 00 BVALU3 BSS 3
129 0040          ORG $0040
130 0040 09      AVG  HEX 09,09
0041 09
131 0042 00      SEQ  BSS 1
132 0043 00      SEG1  BSS 1
133 0044 00      TCNT  BSS 1
134 0045 00      BTCNT BSS 1
135 0046 00      DFLAG BSS 1
136 0047 49      GUIDE HEX 49
137 0048 00      DPTS  BSS 1
138 0049 00      STORE BSS 1
139 004A 00      TEMP  BSS 1
140 004B 00      BTEMP BSS 1
141 004C 00      BYTES BSS 1
142 0200          CRG $0200
143 0200 A0 03    XIRQ  ADR LIRO
144 0202 72 04    XNMI  ADR LNMI
145 *
146 *
147 *      THE INITIALIZATION ROUTINE BEGINS HERE.
148 *
149 *
150 0204 78      SEI          DISABLE SOFTWARE INTERRUPTS
151 0205 A9 02    LDA  =IXSO

```

ORIGINAL PAGE
OF POOR QUALITY

```

152 0207 8D 03 30      STA HCNTRL  SET HARDWARE INT OFF
153 020A AD 00 02      LDA XIRQ
154 020D 8D FE 17      STA IRQL
155 0210 AD 01 02      LDA XIRQ+1
156 0213 8D FF 17      STA IRQL+1
157 0216 AD 02 02      LDA XNMI
158 0219 8D FA 17      STA NMIL
159 021C AD 03 02      LDA XNMI+1
160 021F 8D FB 17      STA NMIL+1
161 0222 A7 BF          LDA =1BF
162 0224 8D 03 17      STA DPIA2
163 0227 A9 7F          LDA =17F
164 0229 8D 01 17      STA DPIA1
165 022C A2 FF          LDX =1FF
166 022E 9A            TXS          SET STACK POINTER TO TOP
167 022F A9 09          LDA =109
168 0231 85 40          STA AVG
169 0233 85 41          STA AVG+1
170 0235 A9 0E          LDA =10E
171 0237 85 19          STA ADATA+1
172 0239 A9 00          LDA =00
173 023B 85 18          STA ADATA
174 023D A2 03          LDX =03
175 023F A0 00          LDY =00
176 0241 91 18          LOOP14 STA (ADATA),Y  CLEAR BYTE
177 0243 C8            INY          INCREMENT INDEX
178 0244 D0 FB          BNE LOOP14  JUMP TILL DONE
179 0246 E6 19          INC ADATA+1  BUMP ADDRESS
180 0248 CA            CEX          TEST OUTSIDE LOOP
181 0249 10 F6          RPL LOOP14  JUMP TILL DONE
182 024B A9 0E          LDA =10E
183 024D 85 19          STA ADATA+1
184 024F A9 00          LDA =00
185 0251 85 18          STA ADATA
186 0253 A2 14          LDX =20
187 0255 95 20          LOOP10 STA CLK,X
188 0257 CA            DEX
189 0258 10 FB          RPL LOOP10
190 025A A2 09          LDX =109
191 025C 95 42          LOOP11 STA SEQ,X
192 025E CA            DEX
193 025F 10 FB          RPL LOOP11
194 0261 A9 49          LDA =149
195 0263 85 47          STA GUIDE
196 0265 AD 00 30      LDA INPUT
197 0268 A9 06          LDA =IOS0
198 026A 8D 03 30      STA HCNTRL
199 026D 54            CLI
200
201
202
203
204
205 026E A5 46          BAKRND LDA DFLAG
206 0270 10 26          BPL CONT5  JUMP IF NMI NOT SET

```

*
 *
 * BACKGROUND PROCESSING BEGINS HERE
 *
 *

207	0272	29	7F		AND = \$7F	CLEAR NMI FLAG
208	0274	85	46		STA DFLAG	RETURN FLAGS
209	0276	A9	18		LDY = XDATA	
21	0278	A2	04		LDX = XCLK	
211	027A	20	86	04	JSP SETUP2	SET UP DATA AND CLOCK
212	027D	A9	08		LDA = 08	
213	027F	20	F1	04	JSR XFER	MOVE POSITION TO DATA STORAGE
214	0282	F8			SED	SET DECIMAL MODE
215	0283	18			CLC	CLEAR CARRY
216	0284	A5	48		LDA DPTS	GET DATA POINTS COUNT
217	0286	69	01		ADC = 01	ADD ONE
218	0288	85	48		STA DPTS	SAVE NEW DATA POINT COUNT
219	028A	D8			CLD	CLEAR DECIMAL MODE
220	028B	A5	18		LDA ADATA	GET DATA ADDRESS POINTER
221	028D	18			CLC	
222	028E	69	09		ADC = 09	
223	0290	85	18		STA ADATA	
224	0292	A5	19		LDA ADATA+1	
225	0294	69	00		ADC = 00	
226	0296	85	19		STA ADATA+1	
227	0298	A5	46	CONT5	LDA DFLAG	GET FLAGS
228	029A	29	10		AND = \$10	MASK DPT BIT
229	029C	F0	0D		BEQ CONT5	JUMP IF RESET
230	029E	A9	0C		LDA = 00	
231	02A0	85	F9		STA DISP	
232	02A2	85	FA		STA DISP+1	
233	02A4	A5	42		LDA DPTS	GET DATA POINT COUNTER
234	02A6	85	FB		STA DISP+2	DISPLAY DATA POINTS USED
235	02A8	4C	CC	02	JMP CONT8	
236	02AB	A5	46	CONT6	LDA DFLAG	GET FLAGS
237	02AD	29	40		AND = \$40	MASK TDA
238	02AF	F0	0F		BEQ CONT7	JUMP IF RESET
239	02B1	A2	08		LDX = XTDB	
240	02B3	A0	0A		LDY = XDISP	
241	02B5	20	86	04	JSR SETUP2	SETUP TDB OUTPUT
242	02B8	A9	02		LDA = 02	
243	02BA	20	F1	04	JSR XFER	MOVE TDB TO DISP
244	02BD	4C	CC	02	JMP CONT9	
245	02C0	A2	06	CONT7	LDX = XTDA	
246	02C2	A0	0A		LDY = XDISP	
247	02C4	20	86	04	JSR SETUP2	SET UP ADDRESSES
248	02C7	A9	02		LDA = 02	SET BYTE COUNT
249	02C9	20	F1	04	JSR XFER	MOVE TDA TO DISPLAY
250	02CC	A5	45	CONT8	LDA BTCNT	GET GRI COUNT
251	02CE	C9	64		CMP = 100	
252	02D0	D0	0C		BNE CONT9	JUMP IF EQUAL
253	02D2	A9	0D		LDA = 00	
254	02D4	85	45		STA BTCNT	CLEAR COUNT
255	02D6	A5	46		LDA DFLAG	GET FLAGS
256	02D8	49	FF		EOR = \$FF	COMPLEMENT
257	02DA	49	0F		EOR = \$0F	COMPLEMENT MUX BIT
258	02DC	85	46		STA DFLAG	SAVE BACK TO FLAGS
259	02DE	A5	46	CONT9	LDA DFLAG	GET FLAGS
260	02E0	29	20		AND = \$20	
261	02E2	F0	12		BEQ CONT10	BRANCH IF RESET

ORIGINAL PAGE IS
OF POOR QUALITY

207	0272	29	7F	AND = \$7F	CLEAR NMI FLAG
208	0274	85	46	STA DFLAG	RETURN FLAGS
209	0276	40	18	LDY = XDATA	
210	0278	A2	04	LDX = XCLK	
211	027A	20	86 04	JSP SETUP2	SET UP DATA AND CLOCK
212	027D	A9	08	LDA = 08	
213	027F	20	F1 04	JSR XFER	MOVE POSITION TO DATA STORAGE
214	0282	F8		SED	SET DECIMAL MODE
215	0283	18		CLC	CLEAR CARRY
216	0284	A5	48	LDA DPTS	GET DATA POINTS COUNT
217	0286	69	01	ADC = 01	ADD ONE
218	0288	85	48	STA DPTS	SAVE NEW DATA POINT COUNT
219	028A	D8		CLD	CLEAR DECIMAL MODE
220	028B	A5	18	LDA ADATA	GET DATA ADDRESS POINTER
221	028D	18		CLC	
222	028E	69	03	ADC = 09	
223	0290	85	18	STA ADATA	
224	0292	A5	19	LDA ADATA+1	
225	0294	69	00	ADC = 00	
226	0296	85	19	STA ADATA+1	
227	0298	A5	46	CONT5 LDA DFLAG	GET FLAGS
228	029A	29	10	AND = \$10	MASK DPT BIT
229	029C	F0	0D	BEQ CONT5	JUMP IF RESET
230	029E	A9	0C	LDA = 00	
231	02A0	85	F9	STA DISP	
232	02A2	85	FA	STA DISP+1	
233	02A4	A5	42	LDA DPTS	GET DATA POINT COUNTER
234	02A6	85	FB	STA DISP+2	DISPLAY DATA POINTS USED
235	02A8	4C	CC 02	JMP CONT8	
236	02AB	A5	46	CONT6 LDA DFLAG	GET FLAGS
237	02AD	29	40	AND = \$40	MASK TDA
238	02AF	F0	0F	BEQ CONT7	JUMP IF RESET
239	02B1	A2	08	LDX = XTDB	
240	02B3	A0	0A	LDY = XDISP	
241	02B5	20	86 04	JSR SETUP2	SETUP TDB OUTPUT
242	02B8	A9	02	LDA = 02	
243	02BA	20	F1 04	JSR XFER	MOVE TDB TO DISP
244	02BD	4C	CC 02	JMP CONT8	
245	02C0	A2	06	CONT7 LDX = XTDA	
246	02C2	A0	0A	LDY = XDISP	
247	02C4	20	86 04	JSR SETUP2	SET UP ADDRESSES
248	02C7	A9	02	LDA = 02	SET BYTE COUNT
249	02C9	20	F1 04	JSR XFER	MOVE TDA TO DISPLAY
250	02CC	A5	45	CONT8 LDA BTCNT	GET GRI COUNT
251	02CE	C9	64	CMP = 100	
252	02D0	D0	0C	BNE CONT9	JUMP IF EQUAL
253	02D2	A9	00	LDA = 00	
254	02D4	85	45	STA BTCNT	CLEAR COUNT
255	02D6	A5	46	LDA DFLAG	GET FLAGS
256	02D8	49	FF	EOR = \$FF	COMPLEMENT
257	02DA	49	DF	EOR = \$DF	COMPLEMENT MUX BIT
258	02DC	85	46	STA DFLAG	SAVE BACK TO FLAGS
259	02DE	A5	46	CONT9 LDA DFLAG	GET FLAGS
260	02E0	29	20	AND = \$20	
261	02E2	F0	12	BEG CONT10	BRANCH IF RESET

ORIGINAL PAGE IS
OF POOR QUALITY

262	02E4	A5	28		LDA TDB+2	LOW BYTE OF TDB
263	02E6	20	FA	04	JSR BCD	CONVERT FOR OUTPUT
264	02E9	C9	46		MODIFY CMP = \$40	TEST FOR > \$40
265	02EB	90	03		BCC NOCHG	IF < \$40 DO NOT MODIFY
266	02ED	19			CLC	
267	02EE	69	47		ADC = \$40	MODIFY FOR OUTPUT
268	02F0	8D	02	17	NOCHG STA CHART	OUTPUT TO CHART
269	02F3	4C	FE	02	JMP CONT11	
271	02F6	A5	25		CONT10 LDA TDA+2	GET LOW BYTE OF TDA
271	02F8	20	FA	04	JSR BCD	CONVERT FOR OUTPUT
272	02FB	4C	EB	02	JMP MODIFY	
273	02FE	20	6A	1F	CONT11 JSR GETKEY	GET KEYBOARD KEY
274	0301	C9	0A		CMP = \$7A	TEST FOR A
275	0303	00	03		BNE CONT12	JUMP IF NOT A
276	0305	A5	46		LDA DFLAG	GET FLAGS
277	0307	29	AF		AND = \$AF	CLEAR DPT FLAG
278	0309	85	46		STA DFLAG	SAVE TO FLAGS
279	030B	4C	49	03	JMP CONT16	
280	030E	C9	0B		CONT12 CMP = \$08	TEST FOR B
281	0310	00	0B		BNE CONT13	JUMP IF NOT B
282	0312	A5	46		LDA DFLAG	GET FLAGS
283	0314	29	EF		AND = \$EF	CLEAR DPT BIT
284	0316	09	40		ORA = \$40	SET TDA BIT
285	0318	85	46		STA DFLAG	RETURN FLAGS
286	031A	4C	49	03	JMP CONT16	
287	031D	C9	01		CONT13 CMP = 01	TEST FOR 1
288	031F	00	0D		BNE CONT14	JUMP IF NOT 1
289	0321	A5	25		LDA TDA+2	GET LSB TDA
290	0323	85	47		STA GUIDE	SAVE TO GUIDE WORD
291	0325	A5	46		LDA DFLAG	
292	0327	29	F7		AND = \$F7	
293	0329	85	46		STA DFLAG	RESET GUIDE BIT
294	032B	4C	49	03	JMP CONT16	
295	032E	C9	02		CONT14 CMP = 02	TEST FOR 2
296	0330	00	0D		BNE CONT15	JUMP IF NOT 2
297	0332	A5	28		LDA TDB+2	GET LSB TDB
298	0334	85	47		STA GUIDE	SAVE TO GUIDE WORD
299	0336	A5	46		LDA DFLAG	
300	0338	09	0B		ORA = \$08	
301	033A	85	46		STA DFLAG	SET GUIDE BIT
302	033C	4C	49	03	JMP CONT16	
303	033F	C9	0D		CONT15 CMP = \$0D	TEST FOR D KEY
304	0341	00	0E		BNE CONT16	JUMP IF NOT EQUAL TO D
305	0343	A5	46		LDA DFLAG	GET FLAGS
306	0345	09	10		ORA = \$10	SET DPT FLAG
307	0347	85	46		STA DFLAG	RETURN FLAGS
308	0349	A5	46		CONT16 LDA DFLAG	GET FLAGS
309	034B	FA			SED	SET DECIMAL MODE
310	034C	29	0B		AND = \$08	MASK GUIDE BIT
311	034E	FD	0B		BEQ CONT17	JUMP IF RESET
312	0350	3B			SEC	SET CARRY
313	0351	A5	47		LDA GUIDE	GET GUIDE WORD
314	0353	E5	2F		SBC TDB+2	SUBTRACT LSB TDB
315	0355	4C	5D	03	JMP CONT20	
316	0358	A5	47		CONT17 LDA GUIDE	GET GUIDE VALUE

ORIGINAL PAGE IS
OF POOR QUALITY

262	02E4	A5	28		LDA	TDB+2	LOW BYTE OF TDB
263	02E6	20	FA	04	JSR	BCD	CONVERT FOR OUTPUT
264	02E9	C9	40		MODIFY	CMP = \$40	TEST FOR > \$40
265	02EB	90	03		BCC	NOCHG	IF < \$40 DO NOT MODIFY
266	02ED	18			CLC		
267	02EE	69	40		ADC	= \$40	MODIFY FOR OUTPUT
268	02F0	8D	02	17	NOCHG	STA	CHART
269	02F3	4C	FE	02		JMP	CONT11
270	02F6	A5	25		CONT10	LDA	TDA+2
271	02F8	20	FA	04		JSR	BCD
272	02FB	4C	E9	02		JMP	MODIFY
273	02FE	20	6A	1F	CONT11	JSR	GETKEY
274	0301	C9	0A			CMP	= \$3A
275	0303	D0	09			BNE	CONT12
276	0305	A5	46			LDA	DFLAG
277	0307	29	AF			AND	= \$AF
278	0309	85	46			STA	DFLAG
279	030B	4C	49	03		JMP	CONT16
280	030E	C9	0B		CONT12	CMP	= \$0B
281	0310	D0	0B			BNE	CONT13
282	0312	A5	46			LDA	DFLAG
283	0314	29	EF			AND	= \$EF
284	0316	09	40			ORA	= \$40
285	0318	85	46			STA	DFLAG
286	031A	4C	49	03		JMP	CONT16
287	031D	C9	01		CONT13	CMP	= \$01
288	031F	D0	0D			BNE	CONT14
289	0321	A5	25			LDA	TDA+2
290	0323	85	47			STA	GUIDE
291	0325	A5	46			LDA	DFLAG
292	0327	29	F7			AND	= \$F7
293	0329	85	46			STA	DFLAG
294	032B	4C	49	03		JMP	CONT16
295	032E	C9	02		CONT14	CMP	= \$02
296	0330	D0	0D			BNE	CONT15
297	0332	A5	28			LDA	TDB+2
298	0334	85	47			STA	GUIDE
299	0336	A5	46			LDA	DFLAG
300	0338	09	08			ORA	= \$08
301	033A	85	46			STA	DFLAG
302	033C	4C	49	03		JMP	CONT16
303	033F	C9	0D		CONT15	CMP	= \$0D
304	0341	D0	0E			BNE	CONT16
305	0343	A5	46			LDA	DFLAG
306	0345	09	10			ORA	= \$10
307	0347	85	46			STA	DFLAG
308	0349	A5	46		CONT16	LDA	DFLAG
309	034B	F8				SED	
310	034C	29	08			AND	= \$08
311	034E	F0	08			BEQ	CONT17
312	0350	38				SEC	
313	0351	A5	47			LDA	GUIDE
314	0353	E5	28			SBC	TDB+2
315	0355	4C	5D	03		JMP	CONT20
316	0358	A5	47		CONT17	LDA	GUIDE

ORIGINAL PAGE IS
OF POOR QUALITY

317	035A	38		SEC		
318	035B	E5	25	SBC	TDA+2	FORM GUIDENCE
319	035D	C9	20	CONT20	CMP =520	TEST FOR >20
320	035F	20	0E		HCS N1	IF >20 TEST IF >40
321	0361	AA		ADD1	TAX	SET INDEX
322	0362	F0	24		BEQ CENTER	
323	0364	1A		CLC		
324	0365	A9	49		LDA =549	SET CENTER VALUE
325	0367	69	05	LOOP12	ADC =05	
326	0369	CA			DEX	DEC INDEX
327	036A	00	FB		BNE LOOP12	LOOP TILL INDEX ZERO
328	036C	4C	8A	03	JMP CONT18	
329	036F	C9	80	N1	CMP =580	TEST FOR >80
330	0371	90	17		BCC CONT18	IF 20< A <80 JUMP
331	0373	85	48		STA BTEMP	SAVE A
332	0375	38			SEC	SET CARRY
333	0376	A9	00		LDA =00	
334	0378	E5	48		SBC BTEMP	FORM COMPLEMENT
335	037A	AA		SUB	TAX	
336	037B	F0	03		BEQ CENTER	
337	037D	38			SEC	
338	037E	A9	49		LDA =549	SET CENTER VALUE
339	0380	E9	05	LOOP13	SBC =05	
340	0382	CA			DEX	
341	0383	00	FB		BNE LOOP13	LOOP TILL ZERO
342	0385	4C	8A	03	JMP CONT18	
343	0388	A9	49	CENTER	LDA =549	SET CENTER
344	038A	08		CONT18	CLD	CLEAR DECIMAL MODE
345	038B	20	FA	04	JSR BCD	CONVERT TO BINARY
346	038E	8D	00	17	OUT STA METER	WRITE TO METER
347	0391	A5	F9		LDA DISP	XCHG DISPLAY HIGH AND LOW BYTES
348	0393	1A			TAX	
349	0394	A5	FB		LDA DISP+2	
350	0396	85	F9		STA DISP	
351	0398	86	FB		STX DISP+2	
352	039A	2C	1F	1F	JSR SCANDS	SCAN DISPLAY
353	039D	4C	6E	02	JMP BAKRND	LOOP IN BACKGROUND
354				*		
355				*		
356				*	IRQ INTERRUPT SERVICE ROUTINE	
357				*		
358				*		
359	03A0	85	F3	LIRQ	STA SAVEA	SAVE A ACCUM.
360	03A2	68			PLA	GET STATUS
361	03A3	85	F1		STA SAVEP	SAVE STATUS IN IMAGE
362	03A5	29	10		AND =510	MASK BRK BIT
363	03A7	F0	03		BEQ UNIT	JUMP IF NOT SET
364	03A9	4C	05	1C	JMP SAVE1	
365	03AC	A5	F1	UNIT	LDA SAVEP	GET STATUS
366	03AE	48			PHA	RESTORE AS BEFORE INTERRUPT
367	03AF	84	F4		STY SAVEY	SAVE Y AINDEX
368	03B1	86	F5		STX SAVEX	SAVE X INDEX
369	03B3	A9	02		LDA =IXSO	DISABLE INTERRUPTS
370	03B5	8D	03	30	STA HCNTRL	
371	03B8	A5	10		LDA VALUE1	SAVE VALUE1 AND VALUE2

372	038A	48		PHA	
373	038C	A5	11	LDA	VALUE1+1
374	038D	48		PHA	
375	038E	A5	12	LDA	VALUE2
376	038C	48		PHA	
377	03C1	A5	13	LDA	VALUE2+1
378	03C3	48		PHA	
379	03C4	D8		CLD	CLEAR DECIMAL MODE
380	03C5	AD	04 30	LDA	MT8 GET MASTER TEST BIT
381	03C8	30	65	BMI	MASTER IF SET JUMP
382	03CA	A5	43	LDA	SEQ1
383	03CC	C9	02	CMP	=02 TEST FOR INVALID INDEX
384	03CE	F0	56	BEQ	ERROR JUMP IF INVALID
385	03D0	A0	02	LDY	=XINPUT SET UP INPUT POINTER
386	03D2	18		CLC	CLEAR CARRY
387	03D3	A9	14	LDA	=XAVGA SET UP AVERAGE POINTER
388	03D5	65	42	ADC	SEQ ADD SEQUENCE
389	03D7	AA		TAX	READY FOR SETUP ADDRESS
390	03D8	20	86 04	JSR	SETUP2 SET UP THE ADDRESSES
391	03D8	A0	02	LDY	=02
392	03D0	F8		SED	SET DECIMAL MODE
393	03DE	18		CLC	CLEAR CARRY
394	03DF	B1	10	LDA	(VALUE1),Y ADD INPUT VALUE TO AVERAGE
395	03F1	71	12	ADC	(VALUE2),Y
396	03E3	91	10	STA	(VALUE1),Y
397	03E5	88		DEY	
398	03E6	B1	10	LDA	(VALUE1),Y
399	03F8	71	12	ADC	(VALUE2),Y
400	03EA	91	10	STA	(VALUE1),Y
401	03EC	88		DEY	
402	03ED	B1	12	LDA	(VALUE2),Y
403	03EF	29	0F	AND	=0F
404	03F1	71	10	ADC	(VALUE1),Y
405	03F3	91	10	STA	(VALUE1),Y
406	03F5	D3		CLD	CLEAR DECIMAL MODE
407	03F6	A6	43	LDX	SEQ1
408	03F8	D6	40	DEC	AVG,X DEC AVERAGE COUNTER
409	03FA	10	55	BPL	CONT2
410	03FC	A9	02	LDA	=02 SET BYTE COUNT
411	03FE	20	D4 04	JSR	ROUND ROUND AVERAGE VALUE
412	0401	A9	02	LDA	=02 SET BYTE COUNT
413	0403	20	BA 04	JSR	SHIFT DIVIDE BY 10
414	0405	18		CLC	CLEAR CARRY
415	0407	A9	06	LDA	=XTDA BACKGROUND TDA
416	0409	65	42	ADC	SEQ
417	040B	AA		TAX	MOVE TO X
418	040C	20	7D 04	JSR	SETUP3 SET UP TO POINTER
419	040F	A9	02	LDA	=02 SET BYTE COUNT
420	0411	20	F1 04	JSR	XFER MOVE BYTES
421	0414	A9	00	LDA	=00
422	0416	A0	02	LDY	=02
423	0418	91	10	STA	(VALUE1),Y CLEAR AVERAGE VALUE
424	041A	88		DEY	
425	041B	10	FB	BPL	LOOP6 LOOP TIL DONE
426	041D	A6	43	LDX	SEQ1 GET SEQUENCE

427	041F	A9	09		LDA =09	AVG-1 LOOP COUNT
428	0421	95	40		STA AVG,X	SAVE TO PROPER CCOUNTER
429	0423	40	51	04	JMP CONT2	
430	0426	A9	00		ERROR LDA =00	
431	0428	85	42		STA SEQ	CLEAR SEQUENCE
432	042A	85	43		STA SEQ1	
433	042C	40	57	04	JMP CONT3	LOOP BACK
434	042F	A9	00		MASTER LDA =00	
435	0431	85	42		STA SEQ	CLEAR SEQUENCES
436	0433	85	43		STA SEQ1	
437	0435	76	44		INC TCNT	BUMP GRI COUNTER
438	0437	E6	45		INC BTCNT	BUMP BACKGROUND COUNTER
439	0439	A5	44		LDA TCNT	
440	043B	C9	0A		CMP =10	TEST FOR 10
441	043D	D0	18		BNE CONT3	IF NOT 10 JUMP
442	043F	18			CLC	CLEAR CARRY
443	0440	A2	04		LDX =XCLK	SET UP POINTER
444	0442	20	94	04	JSR SETUP1	
445	0445	A9	02		LDA =02	SET BYTE COUNT
446	0447	20	90	04	JSR INCBYT	BUMP TIME
447	044A	A9	00		LDA =00	
448	044C	85	44		STA TCNT	CLEAR GRI COUNTER
449	044E	40	57	04	JMP CONT3	
450	0451	F6	42		CONT2 INC SEQ	
451	0453	E6	42		INC SEQ	INCREMENT SEQ BY 2
452	0455	E6	43		INC SEQ1	INC SEQ1
453	0457	68			CONT3 PLA	RESTORE VALUE1 AND VALUE2
454	0458	85	13		STA VALUE2+1	
455	045A	68			PLA	
456	045B	85	12		STA VALUE2	
457	045D	68			PLA	
458	045E	85	11		STA VALUE1+1	
459	0460	68			PLA	
460	0461	85	10		STA VALUE1	
461	0463	46	F5		LDX SAVEX	RESTORE X
462	0465	A4	F4		LDY SAVEY	RESTORE Y
463	0467	AD	00	30	LDA INPUT	RESET HARDWARE IRQ FLAG
464	046A	A9	06		LDA =IOS0	
465	046C	8D	03	30	STA HCNTRL	ENABLE INTERRUPTS
466	046F	A5	F3		LDA SAVEA	RESTORE A
467	0471	40			RTI	
468					*	
469					*	
470					*	NMI INTERRUPT SERVICE ROUTINE
471					*	
472					*	
473	0472	85	F3		LNMI STA SAVEA	SAVE A ACCUM.
474	0474	A5	46		LDA DFLAG	GET FLAGS
475	0476	C9	80		CRA =380	
476	0478	85	46		STA DFLAG	SET NMI FLAG
477	047A	A5	F3		LDA SAVEA	RESTORE A ACCUM.
478	047C	40			RTI	RETURN
479					*	
480					*	
481					*	

```

482      *      THIS ROUTINE IS USED TO LOAD THE PROPER OPERAND
483      *      ADDRESSES BEFORE ENTERING A SUBROUTINE.
484      *      UPON ENTERING, THE X REG. HAS THE ZERO PAGE ADDRESS
485      *      OF THE LSB FOR VALUE1. THE Y REG. HAS THE ZERO
486      *      PAGE ADDRESS FOR THE LSB OF VALUE2.
487      *
488      *
489      047D B5 00      SETUP3 LDA PORGIN,X
490      047F 85 12      STA VALUE2
491      0481 B5 01      LDA PORGIN+1,X
492      0483 85 13      STA VALUE2+1
493      0485 60      RTS
494      0486 8A      SETUP2 TXA
495      0487 48      PHA          SAVE X INDEX
496      0488 98      TYA
497      0489 AA      TAX          SET UP SECOND INDEX
498      048A B5 00      LDA PORGIN,X GET LSB POINTER SECOND VALUE
499      048C 85 12      STA VALUE2 SAVE POINTER LSB FOR SUBROUTINE
500      048E B5 01      LDA PORGIN+1,X GET MSB POINTER SECOND VALUE
501      0490 85 13      STA VALUE2+1 SAVE POINTER MSB FOR SUBROUTINE
502      0492 68      PLA
503      0493 AA      TAX          RESTORE VALUE OF X INDEX
504      0494 B5 00      SETUP1 LDA PORGIN,X GET LSB POINTER FIRST VALUE
505      0496 85 10      STA VALUE1 SAVE POINTER LSB FOR SUBROUTINE
506      0498 B5 01      LDA PORGIN+1,X GET MSB POINTER FIRST VALUE
507      049A 85 11      STA VALUE1+1 SAVE POINTER MSB FOR SUBROUTINE
508      049C 60      RTS          RETURN
509      *
510      *
511      *      THIS ROUTINE WILL INCREMENT THE VALUE POINTED TO BY THE
512      *      ADDRESS IN VALUE1.
513      *
514      *
515      049D A8      INCRBYT TAY          SET UP INDEX
516      049E F8      SED          SET DECIMAL MODE
517      049F 18      CLC
518      04A0 B1 10      LDA (VALUE1),Y GET FIRST BYTE
519      04A2 69 01      ADC #01      ADD ONE
520      04A4 91 10      STA (VALUE1),Y SAVE VALUE
521      04A6 4C AF 04      JMP CONT1
522      04A9 B1 10      LOOP9 LDA (VALUE1),Y
523      04AB 69 00      ADC #00
524      04AD 91 10      STA (VALUE1),Y
525      04AF 06      CONT1 PHP
526      04B0 88      DEY
527      04B1 30 04      BMI RETUR1
528      04B3 28      PLP
529      04B4 4C A9 04      JMP LOOP9
530      04B7 28      RETUR1 PLP
531      04B8 08      CLD          CLEAR DECIMAL MODE
532      04B9 60      RTS
533      *
534      *      DIVIDE BY 10 ROUTINE
535      *
536      04BA 18      SHIFT CLC          CLEAR CARRY

```

ORIGINAL PAGE IS
OF POOR QUALITY


```

537 048B 85 4A      STA TEMP      SAVE BYTE COUNT
538 048D A2 03      LDX =03      SHIFT COUNT-1
539 048F A5 4A      LOOP3 LDA TEMP      GET BYTE COUNT
540 0491 85 4C      STA BYTES      SET BYTE COUNT
541 0493 A0 0C      LDY =0C      INITIALIZE INDEX
542 0495 B1 10      LOOP2 LDA (VALUE1),Y  FETCH BYTE
543 0497 6A          ROR A          SHIFT RIGHT 1
544 0498 91 10      STA (VALUE1),Y  RESTORE BYTE
545 049A C8          INY            BUMP INDEX
546 049B C6 4C      DEC BYTES      DEC BYTE COUNTER
547 049D 10 F8      BPL LOOP2      LOOP TIL NEG.
548 049F CA          DEX            DEC SHIFT COUNTER
549 04A0 18          CLC            CLEAR CARRY
550 04A1 10 EC      BPL LOOP3      LOOP TIL NEG.
551 04A3 60          RTS            RETURN

```

```

552 *
553 *
554 *      THIS ROUTINE WILL ROUND OFF THE
555 *      VALUE POINTED TO BY VALUE1 THE RESULT
556 *      IS RETURNED TO VALUE1. BYTE COUNT-1
557 *      SHOULD BE IN A BEFORE ENTRY.
558 *
559 *

```

```

560 04D4 A8      ROUND TAY            SET BYTE COUNT
561 04D5 B1 10      LDA (VALUE1),Y  GET BYTE
562 04D7 29 0F      AND =0F        MASK TOP HALF
563 04D9 C9 05      CMP =05
564 04DB 30 12      BMI CONT        IF MINUS RETURN
565 04DD F8          SED            SET DECIMAL MODE
566 04DE 18          CLC            CLEAR CARRY
567 04DF B1 10      LDA (VALUE1),Y  GET BYTE
568 04E1 69 10      ADC =010       BUMP TO ROUND OFF
569 04E3 91 10      STA (VALUE1),Y  SAVE BACK
570 04E5 88          DEY            DEC INDEX
571 04E6 B1 10      LOOP4 LDA (VALUE1),Y  GET NEXT BYTE
572 04E8 69 00      ADC =00        ADD CARRY
573 04EA 91 10      STA (VALUE1),Y  SAVE BACK
574 04EC 88          DEY            DEC INDEX
575 04ED 10 F7      BPL LOOP4      QUIT IF INDEX NEG.
576 04EF D8          CONT CLD        CLEAR DECIMAL MODE
577 04F0 60          RTS            RETURN

```

```

578 *
579 *
580 *      THIS ROUTINE WILL MOVE THE CONTENTS POINTED
581 *      TO BY VALUE1 TO LOCATIONS POINTED TO BY VALUE2.
582 *      ON ENTRY A ACCUM SHOULD CONTAIN BYTE COUNT-1.
583 *
584 *

```

```

585 04F1 A8      XFER TAY            SET BYTE COUNT
586 04F2 B1 10      LOOP7 LDA (VALUE1),Y
587 04F4 91 12      STA (VALUE2),Y
588 04F6 88          DEY
589 04F7 10 F0      BPL LOOP7      LOOP TILL DONE
590 04F9 60          RTS            RETURN

```

```

591 *

```

```
592          *
593          *   THIS ROUTINE WILL CONVERT A BYTE FROM BINARY
594          *   TO BCD. BYTE TO BE CONVERTED ENTERS AND
595          *   LEAVES THROUGH A ACCUM.
596          *
597          *
598      BCD    PHA
599          AND = $F0
600          LSR A
601          STA STORE
602          LSR A
603          LSR A
604          CLC
605          ADC STORE
606          STA STORE
607          PLA
608          AND = $0F
609          ADC STORE
610          RTS
611          END
```

END PASS 2: 0 ERRORS

ORIGINAL PAGE IS
OF POOR QUALITY

SYMB	ADDR	DEF	REFERENCES											
AAVGA	001A	117												
AAVGB	001C	119												
ABINPT	000C	110												
ASTDB	000E	111												
ACLK	0004	106												
ADATA	0018	116	171	173	179	183	185	220	223	224	226	176		
ADD1	0361	321												
ADISP	000A	109												
AINPUT	0002	105												
ATDA	0006	107												
ATDB	0008	108												
AVG	0040	130	168	169	408	428								
AVGA	002F	125												
AVGB	0032	126												
BAKRND	026E	205	353											
BCD	04FA	598	263	271	345									
BINPT	0029	123												
BTENT	0045	134	250	254	438									
BTDB	002C	124												
BTEMP	004B	140	331	334										
BVALU1	0014	114												
BVALU2	0016	115												
BVALU3	0038	128												
BYTES	004C	141	540	546										
CENTER	0388	343	322	336										
CHAPT	1702	67	268											
CLK	0020	120	187											
CONT	04EF	576	564											
CONT1	04AF	525	521											
CONT10	02F6	270	261											
CONT11	02FE	273	269											
CONT12	030E	280	275											
CONT13	031D	287	281											
CONT14	032E	295	288											
CONT15	033F	303	296											
CONT16	0349	308	279	286	294	302	304							
CONT17	0358	316	311											
CONT18	038A	344	328	330	342									
CONT2	0451	450	409	429										
CONT20	035D	319	315											
CONT3	0457	453	433	441	449									
CONT5	0298	227	206											
CONT6	02AB	236	229											
CONT7	02C0	245	238											
CONT8	02CC	250	235	244										
CONT9	02DE	259	252											
DFLAG	0046	135	205	208	227	236	255	258	259	276	278	282		
			285	291	293	299	301	305	307	308	474	476		
DISP	00F9	74	231	232	234	347	349	350	351					
OPIA1	1701	64	164											
OPIA2	1703	65	162											
OPTS	0048	137	216	218	233									
ERROR	0426	430	384											

GETKEY	1F6A	62	273																	
GUIDE	0047	136	195	290	298	313	316													
HCNTRL	3003	95	152	198	370	465														
INCDYT	049D	515	446																	
INPUT	3000	63	196	463																
IOSO	0006	70	197	464																
IOSX	0004	71																		
IRQL	17FE	68	154	156																
IXSO	0002	72	151	369																
IXSX	0000	73																		
LIRQ	03A0	359	143																	
LNMI	0472	473	144																	
LOOP10	0255	187	189																	
LOOP11	025C	191	193																	
LOOP12	0367	325	327																	
LOOP13	0380	339	341																	
LOOP14	0241	176	178	181																
LOOP2	04C5	542	547																	
LOOP3	04BF	539	550																	
LOOP4	04E6	571	575																	
LOOP5	0418	423	425																	
LOOP7	04F2	586	589																	
LOOP9	04A9	522	529																	
MASTER	042F	434	381																	
METER	1700	66	346																	
MODIFY	02E9	264	272																	
NTB	3004	94	380																	
NMIL	17FA	69	158	160																
NOCHG	02F0	268	265																	
N1	036F	329	320																	
OUT	038E	346																		
PORSIN	0000	104	489	491	498	500	504	506												
RETUR1	04B7	530	527																	
ROUND	04D4	560	411																	
SAVEA	00F3	75	359	466	473	477														
SAVEP	00F1	76	361	365																
SAVEX	00F5	77	368	461																
SAVEY	00F4	78	367	462																
SAVE1	1C05	79	364																	
SCANDS	1F1F	61	352																	
SEQ	0042	131	191	388	416	431	455	450	451											
SEQ1	0043	132	382	407	426	432	436	452												
SETUP1	0494	504	444																	
SETUP2	0486	494	211	241	247	390														
SETUP3	047D	489	418																	
SHIFT	048A	536	413																	
STORE	0049	138	601	605	606	609														
SUB	037A	335																		
TCNT	0044	133	437	439	448															
TDA	0023	121	270	289	318															
TDB	0026	122	262	297	314															
TEMP	004A	139	537	539																
UNIT	03AC	365	363																	
VALUE1	0010	112	371	373	458	460	505	507	394	396	398	400								
			404	405	423	518	520	522	524	542	544	561								

ORIGINAL PAGE IS
OF POOR QUALITY

SYMBOL	C	567	569	571	573	586	492	499	501	395	399
VALUE2 0012	113	375	377	454	456	490					
		402	587								
VALUE3 0035	127										
XAVGA 001A	92	327									
XAVGB 001C	93										
XBINPT 000C	85										
XBTDB 000E	86										
XBVAL1 0014	89										
XBVAL2 0016	90										
XCLK 0004	81	210	443								
XDATA 0018	91	209									
XDISP 000A	84	247	246								
XFER 04F1	585	213	243	249	420						
XINPUT 0002	80	385									
XIRQ 0200	143	153	155								
XNMI 0202	144	157	159								
XTDA 0006	82	245	415								
XTDB 0008	83	239									
XVALU1 0010	87										
XVALU2 0012	88										

ORIGINAL PAGE
OF POOR QUALITY

APPENDIX C. Software Subroutines.

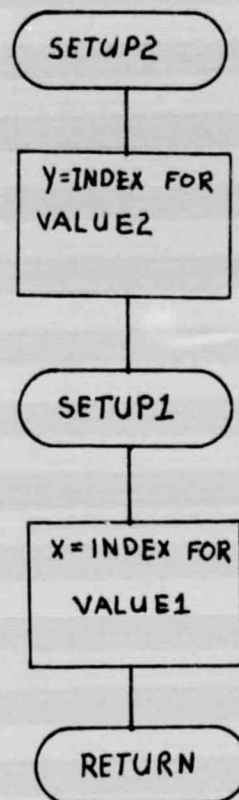
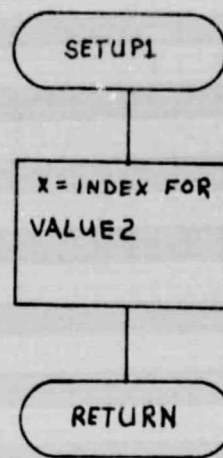


Figure C-1. Pointer Address Setup Flow Diagram.

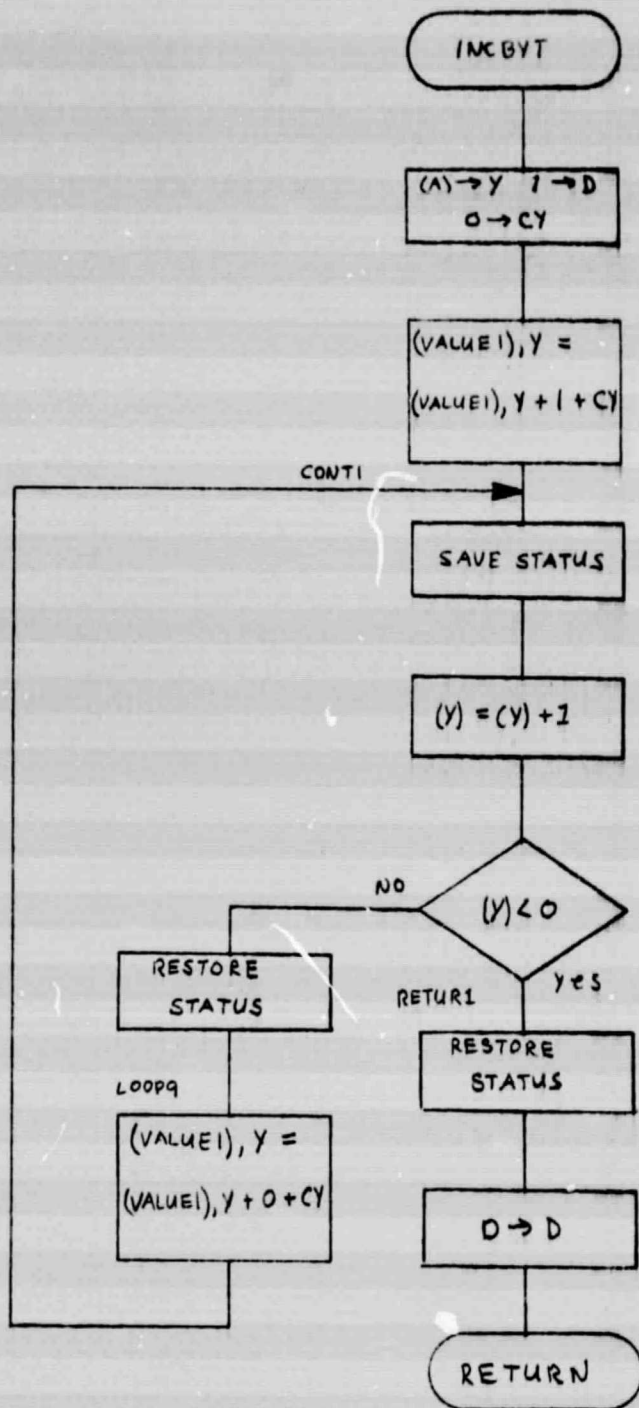


Figure C-2. Increment Byte Flow Diagram.

ORIGINAL PAGE IS
OF POOR QUALITY

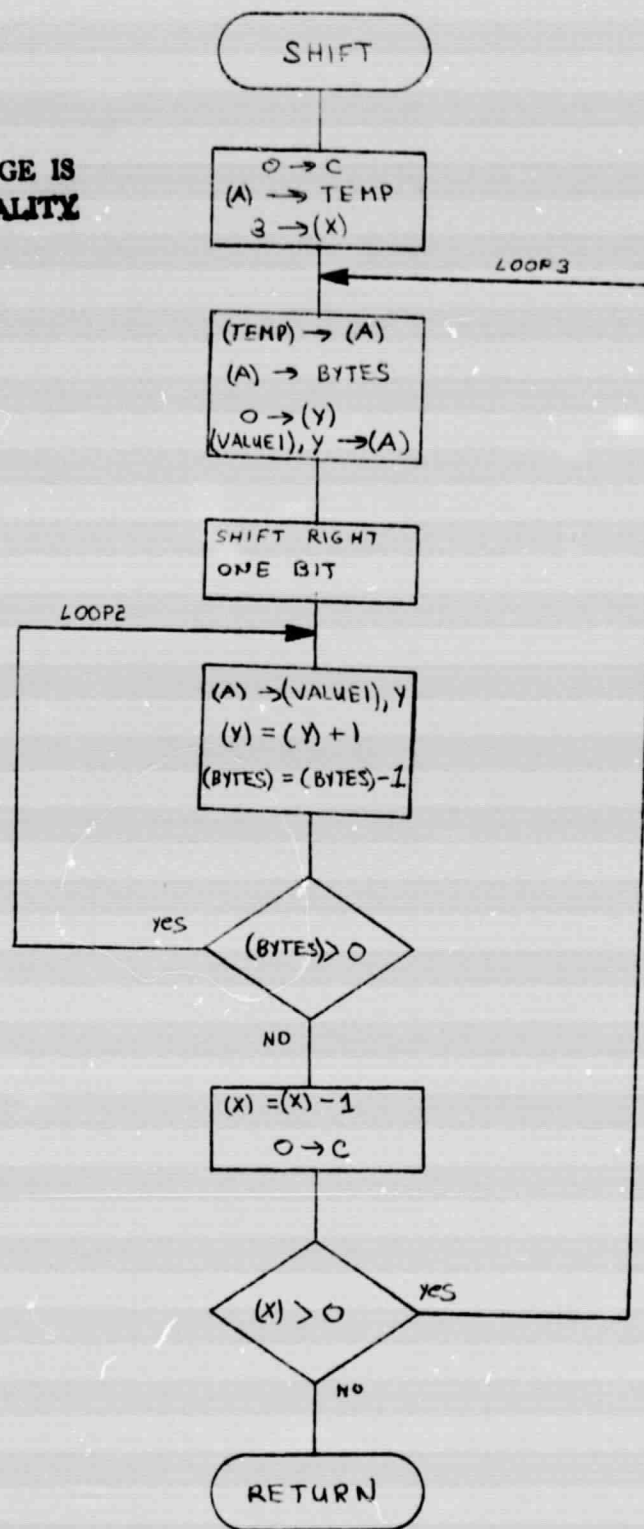


Figure C-3. BCD Shift Flow Diagram.

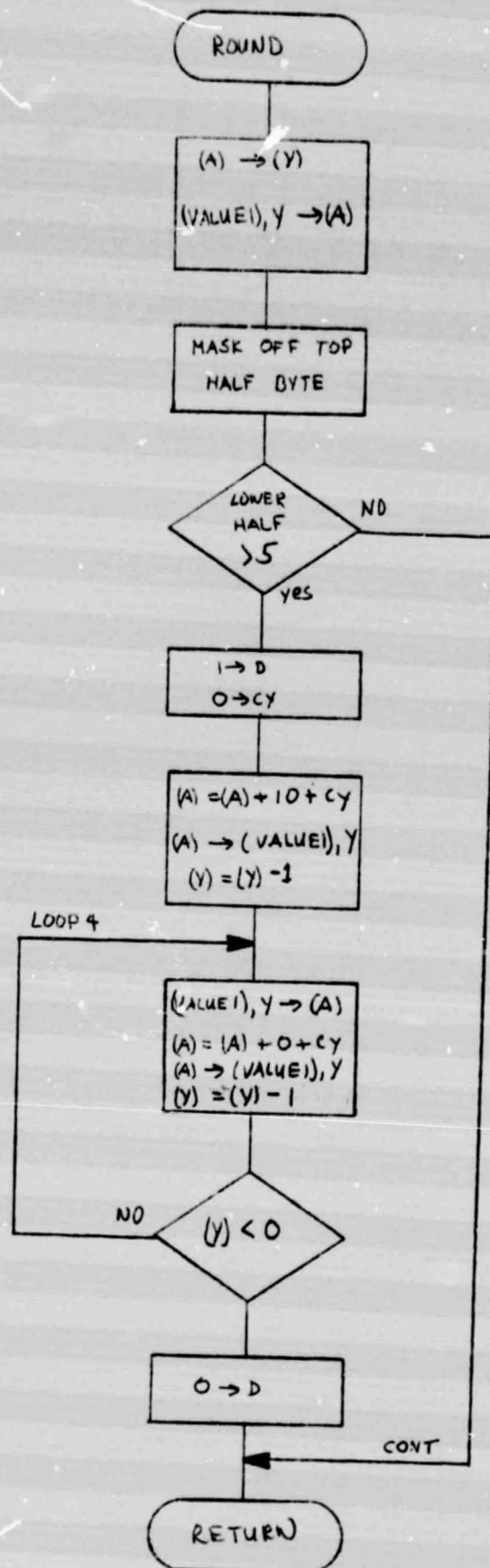


Figure C-4. Least Significant Digit Rounding Flow Diagram.

BCD

ORIGINAL PAGE IS
OF POOR QUALITY

(A) → STACK

MASK BOTTOM
HALF (A)

(A) → STORE

RIGHT SHIFT (A)
2 BITS

0 → CY
(A) = (A) + STORE + CY
(A) → STORE

STACK → (A)
MASK TOP
(A) = (A) + STORE + CY

RETURN

Figure C-5. BCD to Binary Conversion Flow Diagram.

ORIGINAL PRICE IS
OF POOR QUALITY

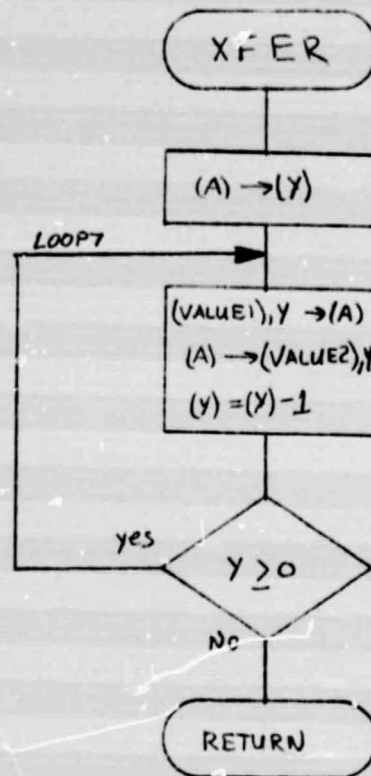


Figure C-6. Transfer Data Flow Diagram.